基于局部并行搜索的分布式约束优化算法框架 *

石美凤，杨　海†，陈　媛，肖诗川，廖　鑫，何　颖

(重庆理工大学 计算机科学与工程学院，重庆 400054)

摘　要：针对当前局部搜索算法在求解大规模、高密度的分布式约束优化问题(DCOPs)时，求解困难且难以跳出局部最优取得进一步优化等问题，提出一种基于局部并行搜索的分布式约束优化算法框架(LPOS)，算法中 Agent 通过自身的取值并行地搜索局部所有邻居取值来进一步扩大对解空间的搜索，从而避免算法过早陷入局部最优。为了保证算法的收敛性与稳定性，设计了一种自适应平衡因子 K 来平衡算法对解的开发和继承能力。并在理论层面证明了并行搜索优化算法可以扩大对解空间的搜索，自适应平衡因子 K 可以实现平衡目的。综合实验结果表明，基于该算法框架的算法在求解低密度和高密度 DCOPs 时性能都优于目前最新的算法。特别是在求解高密度 DCOPs 中有显著的提升。

关键词：分布式约束优化问题；多智能体系统；局部搜索算法；并行搜索优化

中图分类号：TP13　　doi: 10.19734/j.issn.1001-3695.2022.02.0035

Local parallel search framework for distributed constraint optimization problems

Shi Meifeng, Yang Hai†, Chen Yuan, Xiao Shichuan, Liao Xin, He Ying

(*College of Computer Science & Engineering*, *Chongqing University of Technology*, *chongqing* 400054, *China*)

Abstract: In order to solve the problem that it is difficult to get out of the local optimum to achieve further optimization in large-scale dense Distributed Constraint Optimization Problems(DCOPs) . This paper proposed a local parallel search framework(LPOS) for solving algorithms of DCOPs. In LPOS, the Agent searches all the value assignments of its local neighbors in parallel according to its current value assignment to further expand the search of solution space, so as to avoid the algorithm falling into local optimum prematurely. In order to ensure the convergence of the algorithm, this paper designed an adaptive equilibrium factor K to balance the exploration and exploitation ability of the DCOPs solving algorithms. At the theoretical analysis, it proved that the LPOS can expand the search of solution space and the K can achieve the balance. The experimental results demonstrate that the performance of the proposed LPOS is better than the-state-of-the-art algorithms in both low density and high density. Specifically, LPOS significantly superior to the competitors when solving high density DCOPs.

Key words: distributed constrained optimization problem; multi-agent system; local search algorithm; local parallel optimization

0　引言

随着分布式协同运作的人工智能的出现，多智能体系统(MAS)[1]已经成为人工智能的一个重要分支。而分布式约束优化问题(Distributed Constraint Optimization Problems，DCOP)[2]作为智能体协同调度的一个重要框架，是解决分布式智能系统建模和目标之间协同优化的关键技术。DCOP 强调通过局部智能体之间的信息交互以达到全局最优，且具有很强的容错性以及很高的并行性，适用于求解规模大、难度高的组合问题，以分布并行计算的方式快速有效地解决这些问题。目前在任务调度[3]、传感器网络[4,5]、微电网优化控制等领域得到了广泛的应用。然而，由于 DCOPs 是 NP 难问题[6,7]，随着问题规模和复杂程度的增加，完备算法所产生的指数级通信和时间而导致了其应用的局限性。而非完备算法则是通过放弃找到问题的最优解，通过较少的通信和时间成本来寻找次优解，如此便能够很好的应用于大规模的分布式应用模型问题。因此，在近几年非完备算法备受青睐。

非完备算法通常包括以下三类：基于局部搜索的算法[8~12]、基于推理的算法[13~18]和基于采样的算法[19~22]。而局部搜索算法是 DCOPs 中最流行的非完备算法，其每个 Agent 都是根据局部约束条件和从所有相邻 Agent 接收到的信息进行优化的，例如 DSA[23]、DBA[24]和 MGM[25]等。然而，局部搜索算法又具有容易陷入局部最优的缺点。近年来，Chen 等人提出一种基于遗传算法(GA-based framework, LSGA)[26]的框架，利用遗传算子的能力改进局部搜索算法。但交叉算子在优化过程中破坏了交叉块中 Agent 与邻居之间的协调性，使得 Agent 只能依靠自身的局部信息(局部代价以及取值频次)指标来衡量当前解的优劣，仍然不能避免算法陷入局部最优。而 Yu 等人提出了一种局部决策(Partial Decision Scheme, PDS)[27]的框架，该算法框架通过忽略其中一个邻居来搜索有希望的决策分区，从而提高 Agent 的局部利益。然而在大规模高密度的问题中，Agent 周围会有大量邻居，此时 PDS 起到的效果甚微，难以取得进一步的优化。

因此，针对在大规模高密度配置下，分布式约束优化问

收稿日期：2022-02-10；**修回日期：**2022-03-28　**基金项目：**重庆市教育委员会科学技术研究计划青年项目资助项目(KJQN202001139)；重庆市基础研究与前沿探索项目(cstc2018jcyjAX0287)；重庆理工大学科研启动基金资助项目(2019ZD03)；重庆理工大学研究生创新项目(clgycx20202094)

作者简介：石美凤(1989-)，女，重庆人，讲师，博士，主要研究方向为人工智能及应用、多目标优化；杨海(1995-)，男(通信作者)，重庆人，硕士，主要研究方向为人工智能及应用(teachmiss@163.com)；陈媛(1966-)，女，重庆人，教授，硕导，硕士，主要研究方向为数据挖掘；肖诗川(1996-)，女，四川成都人，硕士，主要研究方向为人工智能及应用；廖鑫(1996-)，男，四川达州人，主要研究方向为人工智能及应用；何颖(1988-)，男，重庆人，博士，主要研究方向为图像处理.

题难以被进一步优化的问题，本文提出了一种基于局部并行搜索策略(Locall Parallel Optimization Search, LPOS)的分布式约束优化算法框架，该框架根据 Agent 自身取值发送给周围所有邻居并行地同时搜索自身值域以寻找到最小代价取值来进一步扩大算法对解空间的探索，从而提升算法解的质量。

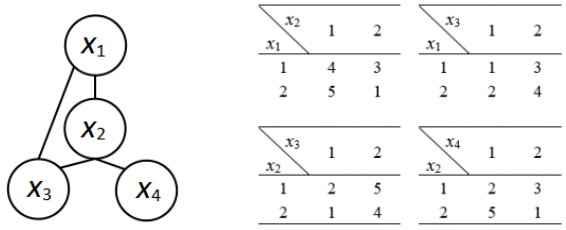
1　介绍

1.1　分布式约束优化问题

分布式约束优化问题(Distributed Constraint Optimization Problems, DCOP)由一个四元组<A，X，D，F>表示，其中：$A=\{a_1,a_2,\cdots a_m\}$ 为 Agent 的集合；$\frac{\text{r\_max-r\_current}}{\text{r\_max}}$ 为变量的集合，并且变量由所属 Agent 进行赋值，每一个 Agent 负责一个或多个变量的取值；$\frac{c_i-c_{\min}}{c_{\max}-c_{\min}}\cdot K<\frac{c_i-c_{\min}}{c_{\max}-c_{\min}}$ 为一个有限值域的集合，Di 为 X 集合中对应变量 xi 的取值值域；$F=\{f_1,f_2,\cdots f_k\}$ 为 Agent 之间约束关系的集合，每一条约束对应着非负映射 $f_i:D_h\times\cdots\times D_k\to R^+$，它包含了当前约束的所有变量组合以及其对应的代价。

DCOP 也可以用约束图来直观地表示。在约束图中每个变量由其对应的 Agent 控制赋值，每个 Agent 仅知道关于控制变量上的约束；各 Agent 在赋值时必须相互协调，使全局目标(如约束函数之和)最优。DCOP 可抽象描述为由节点和边组成的约束图，每个节点对应一个 Agent，边表示节点之间存在约束关系，约束函数表示约束各变量的任意赋值组合的收益(代价)。

如图 1 表示了一个二元约束关系的 DCOP 约束图，其中变量 r_max，每一个 Agent 控制一个变量 X_i；f_{ij} 表示 X_i 和 X_j 之间的约束函数。



(a)约束图　　(b)约束矩阵

图 1　DCOP 实例

Fig. 1　Example of DCOP

由于 Agent 可以控制一个或者多个变量，为了便于理解与讨论，本文假设每个 Agent 只控制一个变量 X_i，它仅知道与该变量相关的约束函数。在 DCOP 中，每个 Agent 与邻居共享约束函数，Agent 之间相互协调找到使 4 个约束函数之和最优的变量赋值，因此 DCOP 的解可以被定义为

$$\sigma^*=\arg\min_{d_i\in D_i}\sum_{f_{ij}\in F}f_{ij}(x_i=d_i,x_j=d_j)\quad(1)$$

1.2　DSA 算法与 MGM2 算法

随机搜索算法(DSA)的基本思想是按照一个预先设定的概率 P 来进行决策。首先，Agent 将控制变量进行随机取值 X_i，并且将取值发送给周围所有邻居，然后各个 Agent 进行重复迭代。在每一次迭代中，Agent 先搜集周围邻居发送的值，然后根据邻居的取值搜索自身值域，并计算取值 X_i 时局部代价与取值 X_j 时局部代价差值 Δ；若 Δ≥0 且随机生成数小于 P，则更新值并发送给周围邻居。通过 DSA 算法能够很快找到次优解，但也极容易陷入局部最优，导致求解质量不好。

而 MGM2 算法是一种在策略上比较保守的算法，通过两个 Agent 组合成联合体的方式进行决策，只有 Agent 之间收益最大的 Agent 才能改变自身取值，否则只能等待下一轮的决策。MGM2 算法与 DSA 算法思想类似，Agent 将控制变量进行随机取值 X_i，然后进行迭代。在每一次迭代中，Agent 先搜集周围邻居发送的值，然后选取能够让 Agent 局部代价降低 Δ_i 最多的值；并将 Δ_i 发送给周围邻居；若 $\Delta>\Delta_i$，则更新值并发送给周围邻居。MGM2 算法虽能在一定程度上改善算法陷入局部最优的问题，但也造成了组合体之间太过关注自身代价而导致算法不能进一步找到更到的解。

1.3　LSGA 框架与 PDS 框架

基于遗传算法的框架[26](LSGA)利用遗传算子(包括选择算子、交叉算子和变异算子)的能力改进局部搜索算法。选择算子由每个个体执行，根据一个适应度函数来评估其基因，并通过适应度函数去取决个体的局部效益和基因值的出现频率。然而由于交叉算子破坏了交叉块中 Agent 与邻居之间的协调性，导致 Agent 只能通过自身局部信息进行评估当前解的质量，因此基于遗传算法框架并不能避免算法陷入局部最优。

而局部决策框(PDS)[27]架是当算法陷入局部最优时，通过 Agent 忽略局部其中一个邻居的策略，以扩大搜索解空间而进一步寻求更优的解。但当面对大规模、高密度的 DCOPs 问题时，仅通过忽略局部一个邻居的策略已难以取得进一步的优化。

因此，本文针对当前局部搜索算法在求解大规模、高密度的 DCOPs 问题中，难以得进一步优化等问题，提出了一种基于局部并行搜索的分布式约束优化算法框架(LPOS)。

2　局部并行搜索优化算法框架算法步骤及实例分析

2.1　算法框架步骤

局部并行搜索优化算法框架

a) 随机搜索算法初始化；

b) 检测 Agent 处于局部最优状态，并计算并行搜索优化概率 POP_i；

c) 生成随机数 $a_i\in(0，1)$；

d) 若 $a_i<POP_i$，则进行并行搜索优化；

e) 随机选择一个邻居 x_b；

f) 将 x_b 取值发送给 Agent，并搜索自身值域选择与 x_b 之间代价最小值 v_i，然后发送给周围所有邻居；

g) 计算 Agent 局部代价增益 g_i；

h) 若 $g_i>0$，则将更新为 Agent 当前取值；

i) 若 $g_i\leqslant 0$，则进一步计算决策优化消息 V*m；

j) 计算全局代价增益 g_c；

k) 若 $g_c>0$，则更新为 Agent 当前取值，否则重复进行 c~j；

根据步骤 b)，当检测到处于局部最优状态(ILB)[27]时，Agent 根据式(6)计算并行优化概率 POP_i 进行局部并行搜索；并行优化概率 POP_i 通过式(2)局部代价水平 L_i 与当前迭代次数水平以及式(7)平衡因子 K 的乘积计算；当求得的代价较大时，局部代价水平会相对较大，且平衡因子 K 也会较大，迭代次数水平仅为控制进程收敛的线性不变函数，故最终的 POP_i 会较大，使得当前 Agent 有更大的概率改变自身的取值，探索求取更优的解，以此降低 POP_i 的值来稳定解的质量。

$$L_i=\frac{c_i-c_{\min}}{c_{\max}-c_{\min}},c_{\max}\neq c_{\min}\quad(2)$$

$$c_i=\sum_{j\in neighbourID}c_{ij}(x_i=a_i,x_j=a_j),\quad a_i\in D_i,a_j\in D_j\quad(3)$$

$$c_{\min}=\min_{a'_i\in D_i}\sum_{j\in neighbourID}c_{ij}\begin{pmatrix}x_i=a'_i,\\ x_j=\arg\min_{a'_j\in D_j}\\ (x_i=a'_i,x_j=a'_j)\end{pmatrix}\quad(4)$$

$$c_{\max}=\max_{a'_i\in D_i}\sum_{j\in neighbourID}c_{ij}\begin{pmatrix}x_i=a'_i,x_j\\=\arg\max_{a'_j\in D_j}\\(x_i=a'_i,x_j=a'_j)\end{pmatrix} \tag{5}$$

其中，a_i 和 a_j 分别是 x_i 和它的邻居 x_j 的当前值。c_{ij} 是 x_i 当前解的局部代价。式(4)与式(5)中，c_{min} 和 c_{max} 分别是 x_i 的局部代价的最大值和最小值，a'_i 的初始值为原始算法处于 ILB 时的取值。neighbourID 是 x_i 周围所有邻居。由式(2)可知，L_i 越大，当前的局部代价越差。

$$POP_i=L_i\cdot\frac{\text{r\_max-r\_current}}{\text{r\_max}}\cdot K \tag{6}$$

$$K=\left(\frac{r\_\max-r\_current}{r\_\max}\right)^{L_i} \tag{7}$$

其中，r_current 和 r_max 分别表示当前迭代次数和最大迭代次数。最大迭代次数被用作局部搜索的终止条件。由式(6)与(7)可知，x_i 的 L_i 越大，局部并行搜索优化的概率越大，POP_i 的影响会随着迭代次数的增加而逐渐减弱。

K 是自适应平衡因子，平衡因子的作用是平衡优化过程中算法的探索和对当前期望解的开发能力。根据算法求解时的代价大小自适应地动态调整 K 值的大小。根据式(7)可知，随着迭代次数的增加，K 值总体趋势逐渐减小，有利于算法收敛与稳定，而局部代价水平的大小使算法能够自适应的调整算法对解的探索与开发的能力。

根据步骤 d)，当 x_i 处于 ILB 以及满足 POP_i 时，进行并行搜索优化。

根据步骤 e)，首先，x_i 随机选择它的一个邻居。然后，x_i 分别通过式(8)~(10)计算新值 a'_i、a'_j 和最小局部代价 $c'_{\min}$。a'_i 和 a'_j 分别是 x_i 和 x_j 对应于 $c'_{\min}$ 的值。

$$a'_i=\arg\min_{i\in D_i}\sum_{\substack{j\in neighbourID\\ j\neq b}}c_{ij}(x_i=a'_i,x_j=a'_j) \tag{8}$$

$$a'_j=\arg\min_{\substack{j\in neighbourID\\ x'_i=a'_i}}c_{ij}(x_i=a'_i,x_j=a'_j) \tag{9}$$

$$c'_{\min}=\sum_{\substack{j\in neighbourID\\ j\neq b}}c_{ij}(x_i=a'_i,x_j=a'_j)+c_{ib}(x_i=a'_i,x_j=x_b) \tag{10}$$

$$g_i=c_i-c'_{\min} \tag{11}$$

根据步骤 g)~步骤 j)，x_i 通过式(11)计算局部代价的增益 g_i，并通过决定是否将其当前值更改为 a'_i；如果 g_i 不大于 0，则进行计算决策信息 V^*_m 与全局代价增益 g_c，若 $g_c<o$ 则意味着 x_i 选择邻居 x_b 进行局部并行搜索优化不能改善其状态，x_i 将保持当前值，然后重复步骤 c)~步骤 j)。

其中，决策信息 V^*_m 根据式(12)改变其值 V^*_k，通过选择邻居 x_b 之外的每个邻居的值来搜索其解空间，从而找到邻居 x_j，使 x_{ij} 的代价最小。然后，x_i 通过式(13)(14)计算 x_i 的新值 a'_i 和全局代价 g_c。

$$V^*_k=\arg\min_{\substack{j\in neighbourID\\ j\neq b}}\sum_{a'_i\in D_i}c_{ij}(V^*_m=a'_i,x_j=a'_j)+\min_{a'_j\in D_j}c_{ij}(V^*_m=a'_i,x_j=a'_j) \tag{12}$$

$$a'_i=\arg\min_{a'_i\in D_i}c_{ij}(x_i=a'_i,x_j=a'_j) \tag{13}$$

$$g_c=\sum_{\substack{j\in neighbourID\\ i\in D_i}}c_{ij}(x_i=a_i,x_j=a_j)-\left(\sum_{\substack{j\in neighbourID\\ j\neq b}}c_{ij}(x_i=a'_i,x_j=a'_j)+c_{ib}(x_i=a'_i,x_j=x_b)\right) \tag{14}$$

2.2　局部并行搜索算法框架实例

以图 1 为例来说明本文所提策略的基本思想。假设所有的 Agent 在多轮之后处于 ILB 状态，且取值分配都为 1，并在下一轮将它们的值 1 发送给它们的邻居。

若此时进行局部决策 PDS，当 x_1 选择邻居 x_3 进行忽略，以寻求打开解空间时，本文发现当 x_1=1 与 x_3=1 时已经是最小值了，已经无法跳出当前取值改变现状。因此本文尝试采用局部并行搜索策略(LPOS)，以迭代次数 800 为终止条件[24]，将其邻居即 neighbourID={x_1,x_2}进行局部并行搜索以达到进一步扩大对解空间的探索，从而寻找到更小的局部代价。也就是说，对于 x_3 的局部而言，通过式(4)(5)分别计算出局部最大值 c_{max}=4+4+5=13 与最小值 c_{min}=1+1+1=3，x_1 与 x_2 同时收到取值 x_3=1，然后分别并行的搜索遍历自身值域，寻找最小代价取值。对于图 1，通过式(3)计算代价，当 x_1=1 时代价为 1，当 x_1=2 时代价为 2，故此时会选择 x_1=1；同时计算当 x_2=1 时代价为 2，当 x_2=2 时代价为 1，故此时会选择 x_2=2。一轮局部并行搜索完成，通过式(8)(9)计算可知最终局部取值为：x_1=1，x_2=2，x_3=1；代价 cost=1+3+1=5，通过式(2)计算 $L_i=\frac{5-1}{13-1}=0.333$，通过式(7) $K=\left(\frac{800-1}{800}\right)^{0.333}=0.999583$，通过式(6)计算可知 $POP_i=0.333*0.99875*0.999583=0.3324$，然后再通过式(10)计算 x_4=2 时最终代价为 5+1=6，根据式(11)计算可知当所有 Agent 取值为 1 时，$g_i=4+1+2+2-6=3>0$；根据算法步骤 d)可知，若随机生成数 $a_i\geqslant POP_i$=0.3324，则将当前所有 Agent 取值为 1 更改为 x_1=1，x_2=2，x_3=1，x_4=2。使得算法能够跳出当前局部最优，进行进一步优化。

3　实验分析

为了验证 LPOS 算法框架的有效性与优越性，本文在随机 DCOP、图着色问题以及无尺度网络问题中与基于 PDS 框架的算法、基于 LSGA 框架的算法以及 GDBA 算法进行性能比较分析。考虑到 DCOP 非完备算法的随机性，实验设计将对每一种模式生成的问题进行 30 次独立求解，并取 30 次统计结果进行对比。为了保证实验的公平性，根据文献[27]，本文对于所有的实验都统一采用 800 次迭代次数作为终止条件，具体实验配置[24,26,27]如表 1 所示。

表 1　实验参数配置

Tab. 1　Experimental parameter configuration

问题类型	值域	问题密度	Agent 数量
随机 DCOP	10	0.1(低) 0.6(高)	150
无尺度网络	10	3(低) 10(高)	150
图着色	3	0.05	200

为了方便对比 LPOS_DSA、LPOS_MGM2 和 PDS_DSA、PDS_MGM2[27] 以及 LSGA_DSA、LSGA_MGM2[27] 和 GDBA[27]算法之间性能的差异，图 2～6 给出了这 7 种算法在三类问题中的收敛曲线图以及表 2、3 分别为 LPOS_DSA 与 LPOS_MGM2 较其他算法在性能上的提升率。

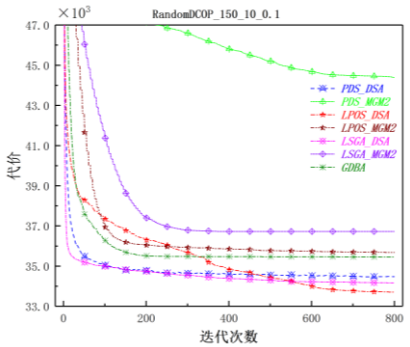


图 2　各算法在低密度随机 DCOP 中收敛曲线图

Fig. 2　Comparison of each algorithm on sparse configuration of random dcops

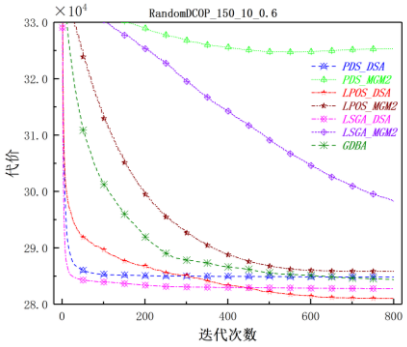


图 3　各算法在高密度随机 DCOP 中收敛曲线图

Fig. 3　Comparison of each algorithm on dense configuration of random dcops

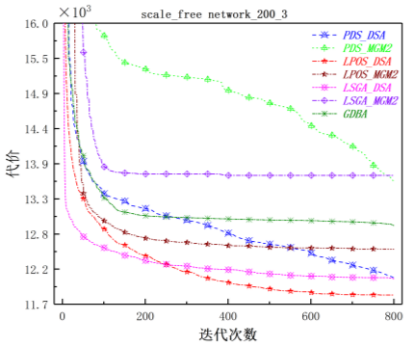


图 4　各算法在低密度无尺度网络中收敛曲线图

Fig. 4　Comparison of each algorithm on sparse configuration of scale-free network

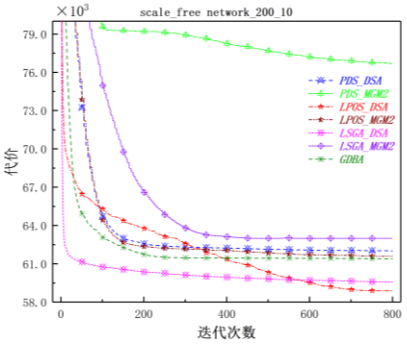


图 5　各算法在高密度无尺度网络中收敛曲线图

Fig. 5　Comparison of each algorithm on dense configuration of scale-free network

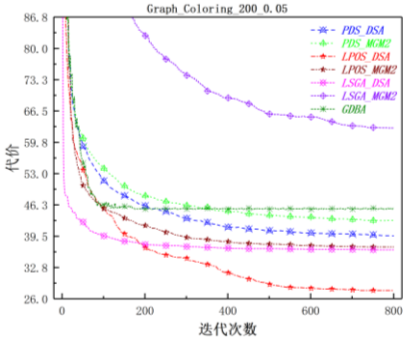


图 6　各算法在图着色问题中收敛曲线图

Fig. 6　Comparison of each algorithm on configuration of graph coloring dcops

从图 2 与 3 可以看出，在随机 DCOP 问题中，在低密度与高密度问题下，PDS_DSA 算法与 LSGA_DSA 算法都以很快的速率下降并随后收敛，导致算法不能够找到更好的解，PDS_MGM2 算法与 LSGA_MGM2 算法由于 MGM2 算法本身采取较为保守的策略进行求解导致在随机 DCOP 问题中表现都比较差；而 LPOS_DSA 算法与 LPOS_MGM2 算法通过并行搜索优化的策略，并采用自适应平衡因子 K 在求解过程中进行平衡解的开发与继承能力，使得 LPOS_DSA 算法与 LPOS_MGM2 算法都能够以较缓的速率下降代价，充分的进行寻求更优的解，以避免快速陷入局部最优。如表二、表三中所示，LPOS_DSA 算法较 PDS_DSA 算法、LSGA_DSA 算法和 GDBA 算法在低密度随机 DCOP 问题中在性能上分别有 2.22%、1.34%和 4.9%的提升，在高密度随机 DCOP 问题中分别有 1.34%、0.62%和 1.17%的提升；而 LPOS_MGM2 算法较 PDS_MGM2 算法与 LSGA_MGM2 算法在低密度随机 DCOP 问题中有 24.55%与 2.85%的提升，而在高密度随机 DCOP 问题中有 12.13%和 4.18%的提升。

在无尺度网络问题中，如图 4、5 所示，在低密度与高密度问题下，LSGA_DSA、LSGA-MGM2 与 GDBA 算法都在迭代次数 150 次左右达到收敛，使得算法不能进一步寻找到更优的解，而 PDS_DSA 在低密度问题中根据其忽略周围其中一个邻居的策略是的算法不断扩大搜索解空间，最终求得比较优秀的解，但对于高密度问题，此策略作用已经非常微小，也在 150 次左右的迭代次数就已经收敛；而 LPOS_DSA 算法与 LPOS_MGM2 算法通过并行搜索优化策略，不断进行解空间的探索，以较缓的速率进行收敛，有效的避免了快速陷入局部最优的问题。最终在表二与表三中可以看出，LPOS_DSA 算法较 PDS_DSA 算法、LSGA_DSA 算法和 GDBA 算法在低密度无尺度网络问题中在性能上分别有 2.22%、8.25%和 8.25%的提升，在高密度无尺度网络问题中分别有 5.05%、4.25%和 4.08%的提升；而 LPOS_MGM2 算法较 PDS_MGM2 算法与 LSGA_MGM2 算法在低密度无尺度网络问题中有 2.22%与 8.25%的提升，且在高密度无尺度网络中有 19.66%和 7.68%的提升。

由于图着色问题是最大约束满足问题，它约束代价取值为 0 或者 1，相比其他问题会简单很多。如图 6 所示，各算法都能够快速收敛，而 LPOS_DSA 算法通过并行搜索优化策略在 200 次迭代后还能继续寻找到明显由于其他算法的解。在表二与表三中可以看出，LPOS_DSA 算法相较于 PDS_DSA 算法、LSGA_DSA 算法和 GDBA 算法在性能上分别有 29.8%、24%和 38.9%的提升；而 LPOS_MGM2 算法较于 PDS_MGM2 算法与 LSGA_MGM2 算法则有 2.24%与 8.26%的提升。

表 2　LPOS_DSA 算法提升率

Tab. 2　Upgrade rate of LPOS_DSA

算法	随机 DCOP		无尺度网络		图着色
	0.1	0.6	3	10	
PDS_DSA	2.22%	1.34%	2.22%	5.05%	29.8%
LSGA_DSA	1.34%	0.62%	8.25%	4.25%	24%
GDBA	4.9%	1.17%	8.25%	4.08%	38.9%

表 3　LPOS_MGM2 算法提升率

Tab. 3　Upgrade rate of LPOS_MGM2

算法	随机 DCOP		无尺度网络		图着色
	0.1	0.6	3	10	
PDS_MGM2	24.6%	12.1%	2.22%	19.7%	2.24%
LSGA_MGM2	2.85%	4.18%	8.25%	7.68%	8.26%

4　结束语

本文提出一种基于局部并行搜索优化的分布式约束优化算法框架(LPOS)。该算法框架通过 Agent 自身取值发送给周围所有邻居并行地同时搜索自身值域来进一步扩大对解空间的搜索。此外，还设计一种自适应的平衡因子 K 以平衡算法在搜索过程中对解的开发与继承能力，有效避免了算法快速陷入局部最优的问题。实验结果表明，与其他算法框架相比，基于局部并行搜索优化算法框架的算法在解的质量上都有较大的提升，特别是在高密度问题中具有更好的效果。未来的

研究工作将通过群体算法的策略应用于局部并行搜索过程中以加快收敛速度；同时，将算法框架应用于连续型分布式约束优化问题也是一个重要的方面。

参考文献：

[1] Pujol G. Multi-agent coordination Dcops and beyond [J]. Proceedings International Joint Conference on Artificial Intelligence, 2011, 22 (3): 28-38.

[2] Mailler R. , Lesser V. Solving distributed constraint optimization problems using cooperative mediation [C]// Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, 2004: 438-445.

[3] 肖宇, 许炜, 程文青. 一种基于分布式约束满足的资源优化模型 [J]. 计算机应用研究, 2009, 26 (3): 864-866. (Xiao Yu, Xu Wei, Cheng Wenqing. Resource allocation optimization model based on DCSP [J]. Application Research of Computers, 2009, 26 (3): 864-866.)

[4] Farinelli A, Rogers A, Jennings N. Agent-based decentralised coordination for sensor networks using the max-sum algorithm [J]. Autonomous agents and multi-agent systems, 2014, 28 (3): 337-380.

[5] Muldoon C, Hare G, Grady M, *et al.* Distributed constraint optimization for resource limited sensor networks [J]. Science of Computer Programming, 2013, 78 (5): 583-593.

[6] Fioretto F, Pontelli E, Yeoh W. Distributed Constraint Optimization Problems and Applications [J]. Artificial Intelligence, 2016, 10 (6): 23-47.

[7] 刘燕丽. 基于冲突的 NP 难问题完备算法的研究 [D]. 湖北: 华中科技大学, 2019.

[8] Deng Yancheng, Chen Ziyu, Chen Dingding, *et al.* AsymDPOP: Complete Inference for Asymmetric Distributed Constraint Optimization Problems. [J]. Trans on Fundamentals of Electronics, 2019, 19 (5): 1182-1189.

[9] Arshad M, Silaghi M. Distributed simulated annealing [C]// Distributed Constraint Problem Solving and Reasoning in Multi-Agent Systems, 2004: 1-12.

[10] Uribe L, Lara A, Deb K, *et al.* A New Gradient Free Local Search Mechanism for Constrained Multi-objective Optimization Problems [J]. Swarm and Evolutionary Computation, 2021, 10 (9): 31-38.

[11] Okamoto S, Zivan R, Nahon, A. Distributed breakout: Beyond satisfaction [C]// In Proceedings of the Twenty-Fifth International Joint Conference on Arificial Intelligence, 2016: 447-453.

[12] 张元恪. 基于局部搜索和遗传策略的网络关键节点算法研究 [D]. 辽宁: 大连理工大学, 2021.

[13] 张文昕. 分布式约束优化问题推理求解算法的可扩展性优化研究 [D]. 重庆: 重庆大学, 2020.

[14] Rogers A, Farinelli A, Stranders R. , *et al.* Bounded approximate decentralised coordination via the max-sum algorithm [J]. Artificial Intelligence, 2011, 17 (5): 730-759.

[15] Rollon E, Larrosa, J. Improved bounded max. sum for distributed constraint optimization [J]. In Principles and practice of constraint programming, 2012: 624-632.

[16] Rollon E, Larrosa J. Decomposing utility functions in bounded max-sum for distributed constraint optimization [J]. In Principles and practice of constraint programming, 2014: 646-654.

[17] Zivan R, Peled H. Max/min-sum distributed constraint optimization through value propagation on an alternating dag [J]. In Proceedings of the 11th International conference on cutonomous agents and multiagent systems, 2012: 265-272.

[18] Chen, Ziyu, Deng, Yancheng, Wu Tengfei, *et al.* A class of iterative refined max sum algorithms via non-consecutive value propagation strategies [J]. Autonomous Agents and Multi-Agent Systems, 2018 (32): 822-860.

[19] Ottens B, Dimitrakakis C, Faltings B. An upper confidence bound approach to distributed constraint optimization problems [J]. Trans on Intelligent Systems and Technology, 2017 (8): 69.

[20] Nguyen D, Yeoh W, Zivan R. A linear-space sampling-based dcop algorithm [J]. Journal of Arificial Intelligence Research, 2019 (64): 705-748.

[21] Zivan R, Okamoto S, Peled H. Explorative anytime local search for distributed constraint optimization [J]. Artificial Intelligence, 2014 (212): 1-26.

[22] 张海鹏, 张扬帆, 孙俊. 基于 Levy 分布的柔软自适应演化采样算法 [J]. 计算机应用研究, 2019, 36 (7): 1994-1997. (Zhang Haipneg, Zhang Yangfan, Sun Jun. Evolutionary sampling approach with soft adaptive Levy probability distribution. [J]. Application Research of Computers, 2019, 36 (7): 1994-1997.)

[23] 余泽鹏. 基于局部搜索的分布式约束优化问题求解算法研究 [D]. 重庆: 重庆大学, 2017.

[24] Hirayama K, Yokoo M. The distributed breakout algorithms [J]. Artificial Intelligence, 2005, 161 (1): 89-115.

[25] Maheswaran R. , Pearce J, Tambe M. A graphical game-based approach [J]. Proceedings of the International Conference on Parallel and Distributed Computing Systems, 2004: 432-439.

[26] Chen Ziyu, Liu Lizhen, He Jingyuan, *et al.* A genetic algorithm based framework for local search algorithms for distributed constraint optimization problems [J]. Autonomous Agents and Multi-Agent Systems, 2020: 34-41.

[27] Yu Zepeng, Chen Ziyu, He Jingyuan. A Partial Decision Scheme for Local Search Algorithms for Distributed Constraint Optimization Problems [C]// The International Conference on Autonomous agents and multi-agent systems, 2017, 175 (2): 730-759.